4732607
5P

The MAP Autonomous Mission Control System

Julie Breed, Steven Coyle, Kevin Blahut, Carolyn Dent, Robert Shendock
Information Systems Center (ISC) Code 580
NASA, Goddard Space Flight Center, Greenbelt MD, USA
Roger Rowe - Altair Aerospace Corporation

Abstract

The Microwave Anisotropy Probe (MAP) mission is the second mission in NASA's Office of Space Science low-cost, Medium-class Explorers (MIDEX) program. The Explorers Program is designed to accomplish frequent, low cost, high quality space science investigations utilizing innovative, streamlined, efficient management, design and operations approaches. The MAP spacecraft will produce an accurate full-sky map of the cosmic microwave background temperature fluctuations with high sensitivity and angular resolution.

The MAP spacecraft is planned for launch in early 2001, and will be staffed by only single-shift operations. During the rest of the time the spacecraft must be operated autonomously, with personnel available only on an on-call basis. Four (4) innovations will work cooperatively to enable a significant reduction in operations costs for the MAP spacecraft. First, the use of a common ground system for Spacecraft Integration and Test (I&T) as well as Operations. Second, the use of Finite State Modeling for intelligent autonomy. Third, the integration of a graphical planning engine to drive the autonomous systems without an intermediate manual step. And fourth, the ability for distributed operations via Web and pager access.

Five (5) off-the-shelf tools have been successfully integrated to provide a new level of autonomy and virtual control. These tools include:

- The Mission Operations Planning and Scheduling System (MOPSS) is an evolving planning and scheduling system, which has been used for a number of Goddard missions. For the MAP mission, MOPSS has been taken to the next level allowing the system to interact with the real-time system to re-plan or change the users defined plans as required for the execution of real-time as well as stored commands. This can then be viewed or modified by the user via a graphical timeline. This timeline drives the autonomous command and control systems, ASIST and Altairis.

- The Advanced System for Integration and Spacecraft Test (ASIST) is a real-time command and control system, which was originally built to support the component development, spacecraft integration, and validation environments. Extensive savings are being realized by using the same system to gather and retain knowledge and expertise from I&T through Operations

- The Altairis Mission Control System (MCS) is a commercial, object-oriented, real-time command and control system which achieves a high degree of intelligent autonomy via Finite State Modeling (FSM). The Altairis MCS is being used to automate everything from arduous test procedures to un-manned health and safety monitoring, and reactive control during lights-out shifts. (A prototype of the Altairis MCS has also been prototyped to run on-board the WIRE spacecraft this year.)

- The Visual Analysis And Graphical Environment (VisAGE) product was used to develop a Web-based interface to the Altairis MCS for MAP. Via this new front end, called Web-Altairis, users can log on from anywhere over their laptop computers to monitor and control the system, and to view graphical representations of parameter trends.

- The Spacecraft Emergency Response System (SERS) is a configurable and robust paging and workflow system based on Lotus COTS groupware. For MAP, SERS two-way paging is being used to allow human-in-the-loop autonomy. SERS is also being used to autonomously generate pass summaries and anomaly reports to expedite problem resolution.

Introduction

In 1989, NASA's Cosmic Background Explorer (COBE) spacecraft was launched, and it subsequently helped to verify the Big Bang Theory. The COBE spacecraft was operated by a round-the-clock staff which included ~20 full-time employees. The Microwave Anisotropy Probe (MAP) spacecraft is planned for launch in early 2001. MAP will pick up where COBE left off by 'MAPping' the entire sky for levels of background radiation in order to determine the center of that Big Bang. Unlike COBE, which was born during the economic boom of the 1980's, MAP will be launched in a cost conscious new century. The MAP spacecraft is planned to be staffed by only 3 full time employees, over a single-shift, Monday through Friday. During the rest of the time, the spacecraft and ground systems must operate autonomously, with personnel available only on an on-call basis.

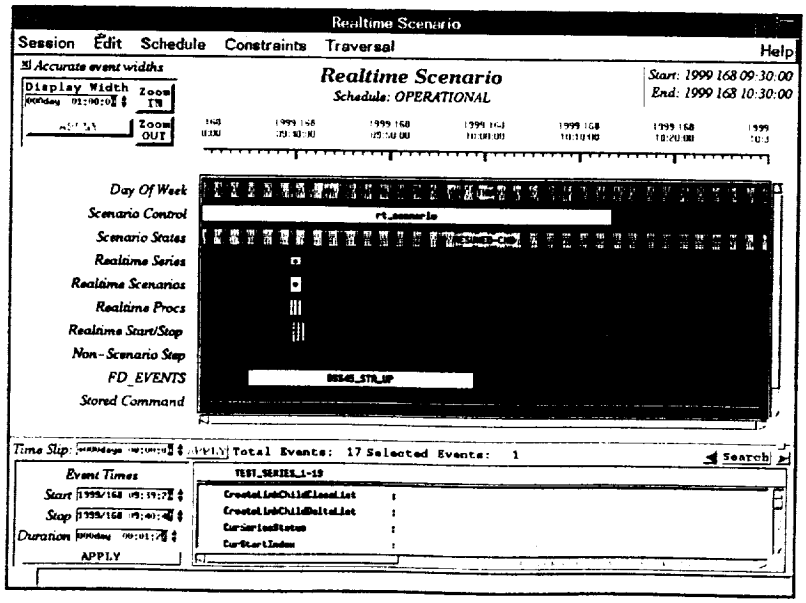
To support MAP's lights-out requirements, five off-the-shelf tools have been successfully integrated. These tools will provide a new level of mission autonomy and virtual control. The automated systems included in MAP's ground system will provide planning, health and safety monitoring, and remote access via the Internet.

Mission Planning as the Driver of Automated Activities

A study to select a mission-planning tool was undertaken by the MAP implementation team in the beginning of the project. The tools considered were Analytical Graphics' GREAS tool, NASA JPL's ASPEN, and GSFC ISC's Mission Operations Planning and Scheduling System (MOPSS). While it was agreed that all three of the tools could handle the basic planning and scheduling needs, the Spacecraft Controller Team (SCT) had a preference for the MOPSS user interface.

MOPSS was developed at the Goddard Space Flight Center (GSFC) and has been used operationally by XTE, TRMM, and Landsat-7. In addition, this year it began supporting the New Millennium Program's Earth Orbiter-1 (EO-1), a spacecraft whose path and activities must be tightly coordinated with the Landsat-7 satellite. MOPSS has been able to handle and automate this coordination. The MOPSS mission legacy has generated a mature product, which has over the years been tuned to operational needs. The screen display is very efficient, and optimizes the number of parallel activities that can be displayed. It is also very configurable in granularity of time and depth of information (fig 1). MOPSS excels as a user-planning tool.

Figure 1: The MOPSS User Interface



For the MAP mission, MOPSS is being used to drive all automated activities. These include stored command procedures as well as real-time ground and space command execution. Users are able to graphically define plans for command execution on a timeline. MOPSS outputs an ASCII file called a Unified Daily Activity Plan (UDAP), which is ingested by the mission control system. The mission control system executes the plan. At the completion of each activity, the mission control system feeds back a completion status message to MOPSS. If the completion is later than planned, MOPSS automatically adjusts subsequent activities based on their constraints and exports a modified UDAP.

In addition to serving as a user planning tool and electronically transferring the user's desired plan to the mission control system, MOPSS is also able to automatically react to changes in environmental conditions. Automated re-planning resulting from delayed activity completion has been mentioned. In addition, MOPSS can automatically re-plan based on changes in network support. Deep Space Network (DSN) schedule changes can be electronically received, and re-planning can occur without human intervention.

A new release of MOPSS provides a Java client user interface. On-call SCT members, as well as PI's are able to log on via the web and view the current mission activity plan and get real-time updates in a read-only fashion.

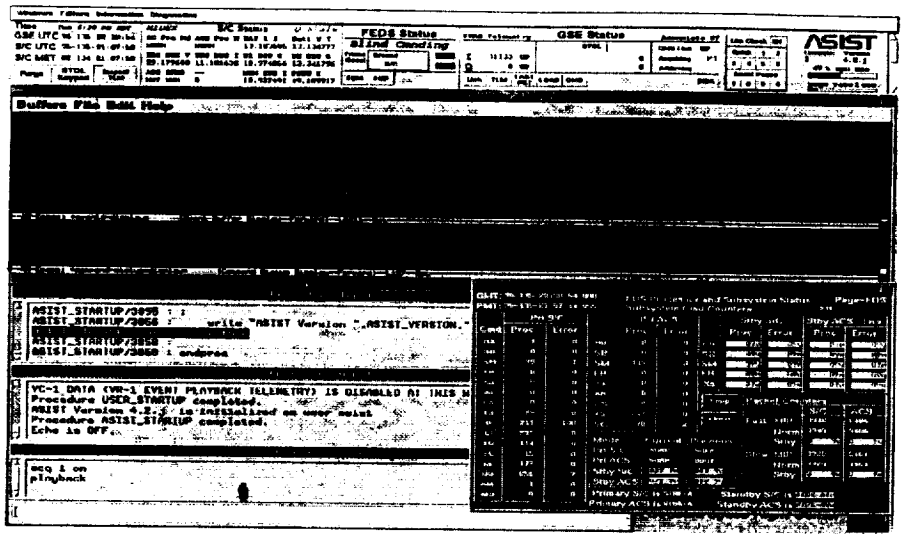
A Common Tool from Spacecraft Integration and Test (I&T) through Operations

The Advanced System for Integration and Spacecraft Test (ASIST) is a real-time command and control system, which was originally built to support component development and spacecraft Integration and Test (I&T). ASIST was not originally intended for use during mission operations. Prior missions, such as COBE, used separate systems and personnel for each phase of the life-cycle. At each turnover there was a significant knowledge transfer effort was required, as the system was effectively thrown over the wall to the next phase. Each new phase involved a learning curve as well as the configuration of a new system. Consequently, at each transition many errors and problems were introduced. Time was lost and costs were escalated.

Because ASIST was originally intended as a testing tool, it is very configurable and capable of handling the additional I&T requirements, such as interfacing with numerous different types of front-end data systems simultaneously. Test tools such as ASIST must provide their users with a high degree of control and internal access. Typical mission control systems shield the operator from the level of detail used in testing, and prevent him/her from inadvertently taking certain actions.

Over its years of supporting various spacecraft in I&T including XTE and TRMM, ASIST has grown into a robust and reliable system capable of supporting mission operations. Other benefits of ASIST include its distributed architecture (fig 2) and its capacity for automation through use of derived parameters and script execution. Every ASIST mission control system includes one ASIST Server, and any number of ASIST Clients. Thus, the data processing load can be distributed, and each operator can customize his/her ASIST workstation as needed. ASIST is also capable of some automation. ASIST handles automation during lights-out shifts for the IMAGE mission, which launched on March 25 of 2000. Operators have written Spacecraft Test and Operations Language (STOL) procedural scripts, which act upon spacecraft parameter values and on values, which are derived from multiple parameters

Figure 2: ASIST User Interface



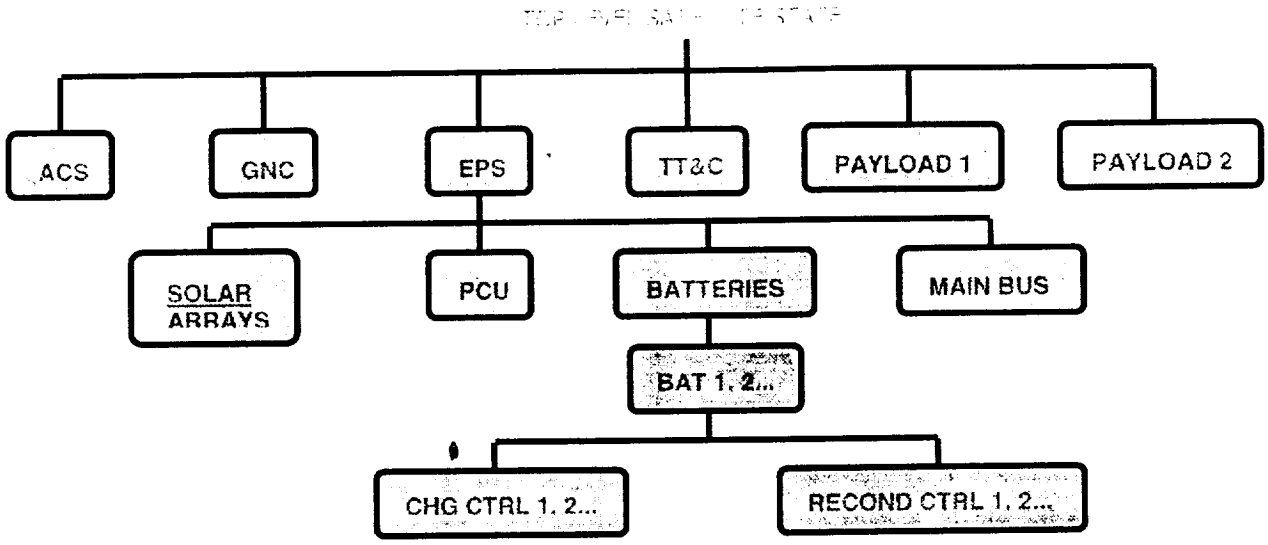
The use of ASIST has been very successful. NASA's GSFC and has saved a substantial amount of time and cost by providing a single environment that meets the varied requirements of each phase of the mission life-cycle.

Finite State Modeling for a Greater Degree of Automation

Although the ASIST ground system is capable of a certain amount of procedural automation, the cost-capped MIDEX mission environment drove the need to further minimize on-going operations costs. Thus, a greater level of autonomy was needed. The project began to study options for adding a more intelligent layer of decision making and control during lights-out shifts. Other missions at Goddard Space Flight Center (GSFC) have experimented with commercial as well as government-developed procedural and rule-based systems. The Altairis Mission Control System (MCS) was selected to support MAP for several key criteria: a modern, commercial, object-oriented system; a hierarchical structure of information management; an intuitive logic paradigm and user interface; and a deterministic approach to intelligent automation.

Finite State Modeling (FSM) is a powerful and compact method of capturing the functionality of complex systems. The system to be modeled is considered as a hierarchy of functional subsystems. At the lowest level, the states of individual components such as relays, batteries, momentum wheels, and valves are determined by their data values. The states of all higher level subsystems are determined by the states of the subsystems or components below them. At the highest level, the states of all lower level subsystems combine to define a single top-level system state. The top-level system may be defined at whatever level is desired. For example, a single satellite, an orbital plane of satellites, a constellation of several orbital planes, or an entire constellation and its associated ground system (fig 3).

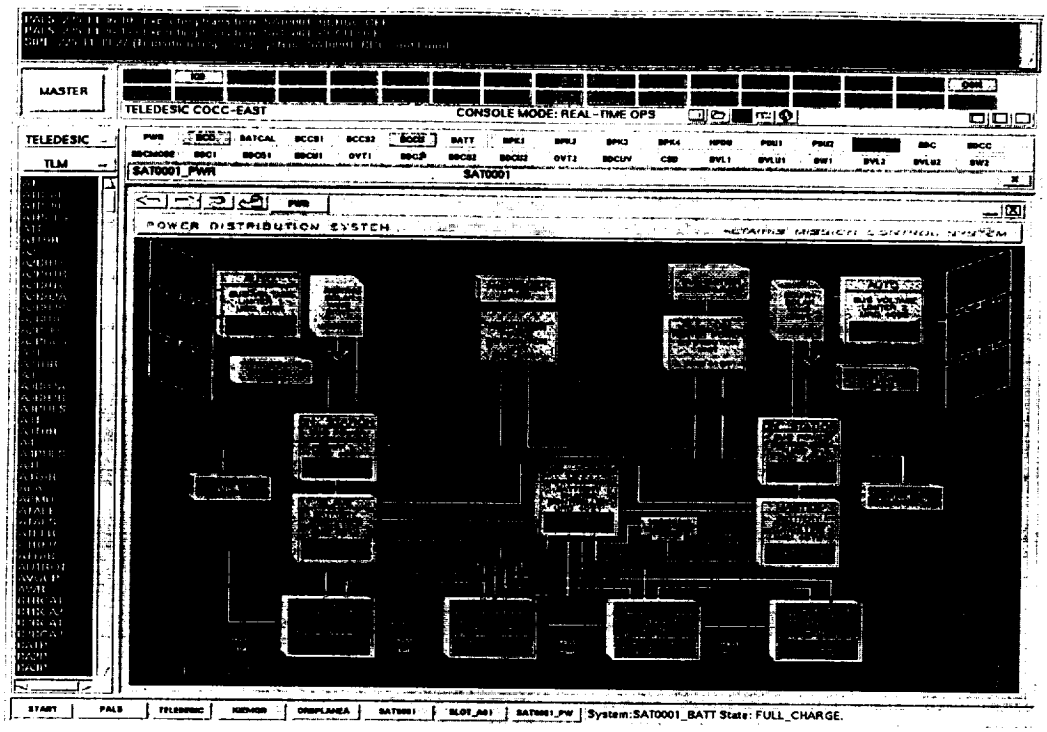
Figure 3: Altairis FSM Hierarchy



Finite State Models are developed directly by domain specialists, and thus the models capture the best existing functional knowledge of the vehicle. Finite State Models can be developed during any phase of the mission, however evidence has shown that having the same system consistently from I&T through Operations simplifies and maximizes the knowledge capture and subsequent it's reuse.

During mission operation, the Altairis software determines the states of components and systems by comparing the incoming data with the predefined expected states (which may include anomalies), and outputting matches. If no matching state is found for any system Altairis will output the state as unrecognized. There are many benefits to this hierarchical structure. First, it enables very concise user displays. Data points are synthesized into informational states, which are communicated in colored lights across the top of the control panel. Operators viewing these high-level status displays can then drill through the structure to reach supporting data (fig 4). Second, it increases performance in terms of throughput. Finite State Modeling can be moved into the ground system front end or even on-board the spacecraft, resulting in output of states rather than individual telemetry points. Third, FSM allows logic and decisions to function at a higher level. State transitions are executed to control the spacecraft based on the current state of a system as well as its desired state at a given time. A subsystem state may include characteristics of hundreds of supporting parameters. Decisions are not simply made based on a range violation of a single parameter.

Figure 4: Altairis MCS User Interface



Traditional mission control systems use single-parameter limit violations to recognize health and safety violations. Limits provide a good first indication of problems, but soon become insufficient. In an operational system, it is nearly impossible to set limits to an acceptable range without diluting alarm response. Limits rarely take the context into consideration. And, even a small number of "acceptable" limit violations can create enough noise to cause the operations crew to miss the one valid violation.

Simply applying a range to a limit for a particular parameter can be a misleading or incomplete indication of the health of a component. Without additional information, it is difficult, if not impossible, to make a correction limit violation determination. For example, during full sun, the parameters of various power system components of a spacecraft will have a value quite different from the period during eclipse or the transition between full sun and eclipse. Values such as bus voltage, current, battery temperature, etc. can not be assessed on an individual basis. The voltage of a battery can be affected by discharge current, recharge current, temperature, and pressure. All of these parameters need to be considered to make a valid determination of the state of the battery.

A Finite State Model (FSM) can accurately model multiple states of a component taking into consideration all of the relevant data. Each datum is considered within the context of all relevant data. The state of a battery, for example, can only be determined by examining the voltage, discharge current, recharge current, temperature, pressure, etc. Any one value might produce a false indicator, but together the values accurately describe the state of the battery. Using FSM, the operator would accurately see the state of a component, so in the above example, the operator would see the system transition from "full sun" to "transition" to "eclipse" without seeing alarms since the states are expected.

Traditional mission control systems typically control the spacecraft through a procedural language such as STOL (Spacecraft Test and Operations Language). As systems become more complex, the a procedural approach becomes unwieldy. Each procedure executes without knowledge of other procedures. Telemetry verification is constrained to only those values relevant to a specific command and do not take the state of the entire system into account. Since each procedure is a self-contained unit, there is repetition of code making maintenance very difficult. If telemetry verification changes, all procedures with that value must be found and changed. Missing even one could have severe consequences.

FSM uses a technique called *state transition* for control. At the lowest level, raw commands are sent to the controlled system; but, in normal operation they are completely hidden from the operator. A state transition will always contain certain attributes including the *entry state* of the system (or the state that it must be in before the transition can be executed), and the *target state* (or the state that it will enter if the transition is successful). A transition may also contain other attributes such as constraints and restrictions on the states of other systems that must be true before it can be executed. In addition, a transition may include instructions for further actions to be taken if the system is not in the correct entry state, or if the transition is or is not successful . A single transition object may encapsulate several pages of conventional scripted procedures.

The use of state transitions for all normal commanding is a very powerful safety feature and entirely precludes the possibility of inadvertently sending an incorrect command and harming the system. Override of this feature is possible for an operator with a sufficiently high access authorization, allowing direct commands to be sent when necessary (such as when in trouble-shooting an anomaly).

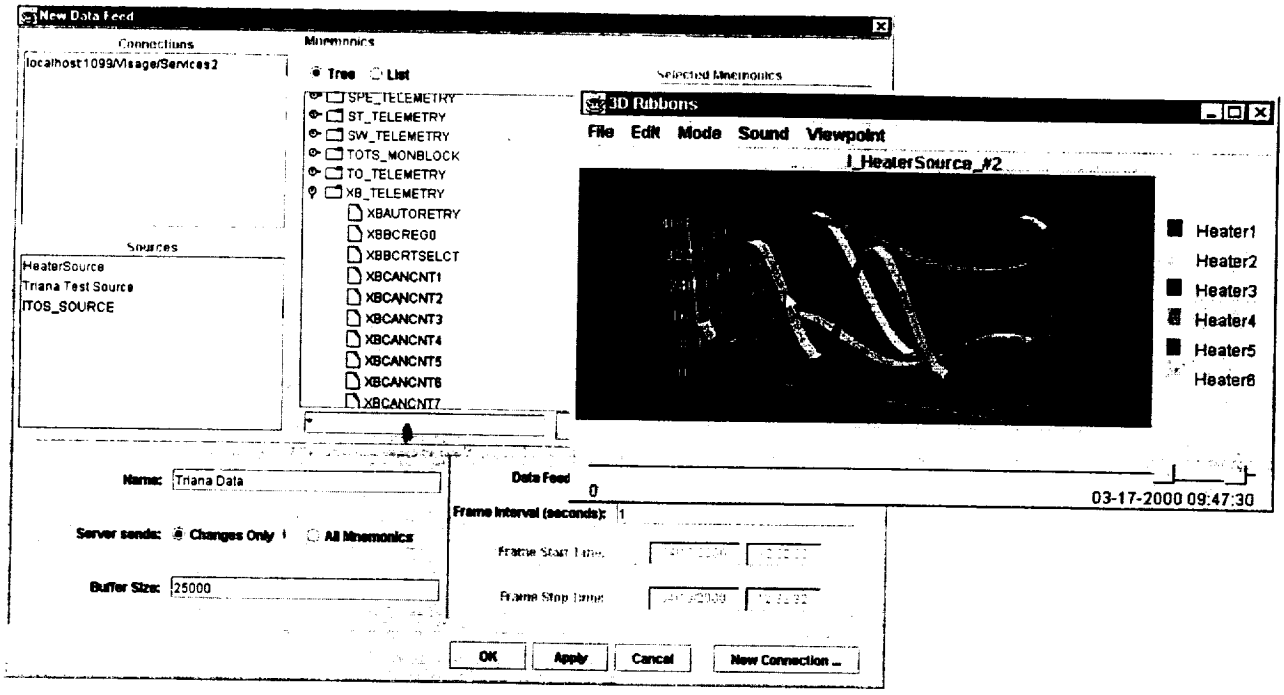
Another benefit of the FSM structure is scalability. The concepts can easily be extended to constellations of hundreds of spacecraft, simply by adding another layer. The hierarchical nature enables efficient displays, which show the top-level status and allow users to drill down for details. Users can implement state transitions at any level in the hierarchy.

FSM systems are also deterministic and easily verifiable. A finite number of states are modeled, one of which is the 'unknown' state. Implementers specify logic and actions for the unknown state as well as the others. This makes FSM ideally suited for on-board execution. GSFC is in the process of implementing the Altairis MCS for an on-board application, and has currently implemented the Finite State Model and Control Engine in <20Kbytes. The proof-of-concept prototype will fly on-board the WIRE spacecraft this year.

A Java Based Tool for Remote Access from Laptop Computers

The Visual Analysis and Graphical Environment (VisAGE) product is a platform-independent tool for engineering data visualization. VisAGE allows users to dynamically select multiple parameters and a visualization style, and then pipes the data to the screen as desired (fig 5).

Figure 5: VisAGE User Interface



VisAGE also includes a tool-kit for rapidly prototyping displays. For MAP, this tool-kit was used to replicate the Altairis status display as a light-weight client. Remote MAP operators will have full access to the Altairis display as well as a range of data visualizations from standard desktop equipment via the internet. In addition, the VisAGE client includes a secure remote commanding capability for selected commands.

Reliable Notification of On-Call Personnel to Reduce the Risk of 'Lights-Out' Ops

The key to ensuring safety and reliability as advanced automation technologies are introduced is the Spacecraft Emergency Response System (SERS). The SERS is a configurable paging and workflow system based on Lotus COTS groupware. The SERS is a modular alert system, which can receive a message from any ground component, and transmit it to a remote user via either Skytel 2-way pagers, email, or telephony. During I&T the SERS is being used to transfer messages from the GSFC weather alert system and several environmental sensors monitoring spacecraft components and facilities. The latest SERS prototype executes in a miniature Web browser on a Palm Pilot screen within a Qualcom telephone.

Because of its Lotus Workflow basis, the SERS is also being used to autonomously generate pass summary and anomaly reports. This eliminates the need for manual generation of reports and log entries, which will greatly expedite the problem resolution process. The SERS provides an on-line history of all events that occur during the lifetime of the spacecraft.

To mitigate the risk of lights-out operations and new technology insertion, the SERS uses 2-way paging and a pre-configured series of actions to ensure that when situations arise, the proper person(s) are notified. Via the 2-way pagers, an on-call SCT member can acknowledge

responsibility for a particular alert. If the primary person does not reply within a configurable amount of time, the SERS will then page the next person on the list, and so on.

In addition, during periods of risk and/or transition, the SERS can be configured to keep remote persons informed of automated procedures, and allow them to approve or inhibit system actions. For example, if the network contact times change, an electronic message is received and the MAP control system can automatically re-plan based on the new information. However, in the early phase of operations, the SERS will issue a page to an on-call SCT member which says, "The MAP mission control system has received a new contact time and is about to re-plan based on that time. Do you approve?" The SCT member will have the option of approving or inhibiting the automated re-plan through their pager. This is just one example of how humans can safely and gradually relinquish control as trust is built in the automated system.

The MAP Autonomous Mission Control System will achieve a high degree of reliability by not only montitoring the status of the spacecraft, but also by monitoring the status of the ground system components themselves. Remote SCT members can be kept informed of nominal automated functions, and will be immediately alerted to any unexpected or problematic situations.

The SERS and VisAGE research and development efforts are each the topic of an additional SpaceOps 2000 paper: "New Human-Computer Interface Concepts for Automation in Mission Operations", and "Web-Altairis: An Internet-Enabled Ground System", respectively.